Problem Set 5: Axiomatic Verification

Hints and Notes

1. Consider the assertion of *weak* correctness: $\{z<0\}$ s $\{y=z+1\}$. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates.

   b. When the initial value of z is -1, the value of y is 17 when s terminates.

   c. When the initial value of z is -3, the program does not terminate.

1. Consider the assertion of *weak* correctness: $\{z<0\}$ s $\{y=z+1\}$. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates. **Wound not:** pre-condition not satisfied

   b. When the initial value of z is -1, the value of y is 17 when s terminates.

   c. When the initial value of z is -3, the program does not terminate.

1. Consider the assertion of *weak* correctness: {z<0} s {y=z+1}. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates. **Wound not:** pre-condition not satisfied

   b. When the initial value of z is -1, the value of y is 17 when s terminates.

   c. When the initial value of z is -3, the program does not terminate.

1. Consider the assertion of *weak* correctness: {z<0} s {y=z+1}. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates. **<u>Wound not</u>:** pre-condition not satisfied

   b. When the initial value of z is -1, the value of y is 17 when s terminates. **<u>Wound not</u>:** Q may or may not hold in this case

   c. When the initial value of z is -3, the program does not terminate.

1. Consider the assertion of *weak* correctness: {z<0} s {y=z+1}. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates. **Wound not:** pre-condition not satisfied

   b. When the initial value of z is -1, the value of y is 17 when s terminates. **Wound not:** Q may or may not hold in this case

   c. When the initial value of z is -3, the program does not terminate.

1. Consider the assertion of *weak* correctness: {z<0} s {y=z+1}. Which of the following observations/facts would allow one to deduce that the assertion is FALSE and which would not? Consider the observations individually and briefly justify your answer for each.

   a. When the initial value of z is 3, the value of y is 4 when s terminates. **Wound not:** pre-condition not satisfied

   b. When the initial value of z is -1, the value of y is 17 when s terminates. **Wound not:** Q may or may not hold in this case

   c. When the initial value of z is -3, the program does not terminate. **Wound not:** *weak* correctness does not require termination

2.

```
{x>y}
  temp := x

  x := y

  y := temp

  if temp>z then
     y := z
     z := temp
        if x>y then
           temp := x
           x := y
           y := temp
        end_if
  end_if
{x≤y≤z}
```

2.

```
{x>y}
   temp := x
{temp=x ∧ x>y}
   x := y

   y := temp

   if temp>z then
      y := z
      z := temp
         if x>y then
            temp := x
            x := y
            y := temp
         end_if
   end_if
{x≤y≤z}
```

2.

```
{x>y}
   temp := x
{temp=x ∧ x>y}
   x := y
{x=y ∧ temp=x’ ∧ x’>y}
   y := temp

   if temp>z then
      y := z
      z := temp
         if x>y then
            temp := x
            x := y
            y := temp
         end_if
   end_if
{x≤y≤z}
```

2.

$\{x>y\}$

```
temp := x
```

$\{temp=x \wedge x>y\}$

```
x := y
```

$\{x=y \wedge temp=x' \wedge x'>y\}$

```
y := temp
```

$\{y=temp \wedge x=y' \wedge temp=x' \wedge x'>y'\} \Rightarrow \{y=temp \wedge temp>x\}$

```
if temp>z then
   y := z
   z := temp
      if x>y then
         temp := x
         x := y
         y := temp
      end_if
end_if
```

$\{x \le y \le z\}$

2.

```
        {x>y}
          temp := x
        {temp=x ∧ x>y}
          x := y
        {x=y ∧ temp=x’ ∧ x’>y}
          y := temp
{y=temp ∧ x=y’ ∧ temp=x’ ∧ x’>y’} => {y=temp ∧ temp>x}
          if temp>z then
            y := z
            z := temp
              if x>y then
      S1        temp := x
                x := y                    S2
                y := temp
              end_if
          end_if
        {x≤y≤z}
```

2. (cont'd)

$\{y=temp \wedge temp>x\}$ if temp>z then S1 $\{x \leq y \leq z\}$

2. (cont'd)

$\{y=temp \land temp>x\}$ if temp>z then S1 $\{x \le y \le z\}$

Using the if-then ROI, we need to show:

(1) $\{y=temp \land temp>x \land temp>z\}$ S1 $\{x \le y \le z\}$?

(2) $(y=temp \land temp>x \land temp \le z) \Rightarrow x<y \le z \Rightarrow Q$ √

2. (cont'd)

$\{y=temp \wedge temp>x\}$ if temp>z then S1 $\{x \le y \le z\}$

Using the if-then ROI, we need to show:

(1) $\{y=temp \wedge temp>x \wedge temp>z\}$ S1 $\{x \le y \le z\}$?

(2) $(y=temp \wedge temp>x \wedge temp \le z) \Rightarrow x<y \le z \Rightarrow Q$ √

For (1) above we have: $\{y=temp \wedge temp>x \wedge temp>z\}$

y := z

z := temp

if x>y then S2

$\{x \le y \le z\}$?

2. (cont'd)

$\{y=temp \wedge temp>x\}$ if temp>z then S1 $\{x \leq y \leq z\}$

Using the if-then ROI, we need to show:

(1) $\{y=temp \wedge temp>x \wedge temp>z\}$ S1 $\{x \leq y \leq z\}$?

(2) $(y=temp \wedge temp>x \wedge temp \leq z) \Rightarrow x<y \leq z \Rightarrow Q$ √

For (1) above we have: $\{y=temp \wedge temp>x \wedge temp>z\}$

y := z

$\{y=z \wedge y'=temp \wedge temp>x \wedge temp>z\}$

z := temp

if x>y then S2

$\{x \leq y \leq z\}$?

2. (cont'd)

$\{y=temp \land temp>x\}$ if temp>z then S1 $\{x \le y \le z\}$

Using the if-then ROI, we need to show:

(1) $\{y=temp \land temp>x \land temp>z\}$ S1 $\{x \le y \le z\}$?

(2) $(y=temp \land temp>x \land temp \le z) \Rightarrow x<y \le z \Rightarrow Q$ √

For (1) above we have: $\{y=temp \land temp>x \land temp>z\}$

y := z

$\{y=z \land y'=temp \land temp>x \land temp>z\}$

z := temp

$\{z=temp \land y=z' \land y'=temp \land temp>x \land temp>z'\} \Rightarrow \{z=temp \land temp>x \land temp>y\}$

if x>y then S2

$\{x \le y \le z\}$?

2. (cont'd)

$\{y=temp \wedge temp>x\}$ if temp>z then S1 $\{x \le y \le z\}$

Using the if-then ROI, we need to show:

(1) $\{y=temp \wedge temp>x \wedge temp>z\}$ S1 $\{x \le y \le z\}$?

(2) $(y=temp \wedge temp>x \wedge temp \le z) \Rightarrow x<y \le z \Rightarrow Q$ √

For (1) above we have: $\{y=temp \wedge temp>x \wedge temp>z\}$

y := z

$\{y=z \wedge y'=temp \wedge temp>x \wedge temp>z\}$

z := temp

$\{z=temp \wedge y=z' \wedge y'=temp \wedge temp>x \wedge temp>z'\} \Rightarrow \{z=temp \wedge temp>x \wedge temp>y\}$

if x>y then S2

$\{x \le y \le z\}$?

for which the if-then ROI may be used a second time.

3. Prove the following assertion using the While-Loop Rule of Inference. Show all steps.

$\{N \geq 1\}$

```
Found := false
Index := N
while (Index>0 & (not Found)) do
   if Key=List[Index] then
      Found := true
   else
      Index := Index-1
   end_if_else
end_while
```

$\{(\text{Found} \wedge \text{Key}=\text{List}[\text{Index}]) \vee$
$(\sim\text{Found} \wedge \forall\ 1 \leq i \leq N \bullet \text{Key} \neq \text{List}[i])\}$

3. Prove the following assertion using the While-Loop Rule of Inference. Show all steps.

$\{N \geq 1\}$

```
Found := false
Index := N
while (Index>0 & (not Found)) do
   if Key=List[Index] then
      Found := true
   else
      Index := Index-1
   end_if_else
end_while
```

$\{(\text{Found} \wedge \text{Key}=\text{List}[\text{Index}]) \vee$
$(\sim\text{Found} \wedge \forall\ 1 \leq i \leq N \bullet \text{Key} \neq \text{List}[i])\}$

What invariant, I, can be used to prove this?

3. Prove the following assertion using the While-Loop Rule of Inference. Show all steps.

$\{N \geq 1\}$

```
Found := false
Index := N
while (Index>0 & (not Found)) do
  if Key=List[Index] then
    Found := true
  else
    Index := Index-1
  end_if_else
end_while
```

$\{(\text{Found} \wedge \text{Key}=\text{List}[\text{Index}]) \vee$
$(\sim\text{Found} \wedge \forall\ 1 \leq i \leq N \bullet \text{Key} \neq \text{List}[i])\}$

$I = (\text{Found} \wedge \ldots) \vee (\sim\text{Found} \wedge \ldots)$

3. Prove the following assertion using the While-Loop Rule of Inference. Show all steps.

$\{N \geq 1\}$

```
Found := false
Index := N
while (Index>0 & (not Found)) do
   if Key=List[Index] then
      Found := true
   else
      Index := Index-1
   end_if_else
end_while
```

$\{(Found \wedge Key=List[Index]) \vee$
$(\sim Found \wedge \forall\ 1 \leq i \leq N \bullet Key \neq List[i])\}$

I = (Found ∧ Key=List[Index]) V
(~Found ∧ …)

3. Prove the following assertion using the While-Loop Rule of Inference. Show all steps.

$\{N \geq 1\}$

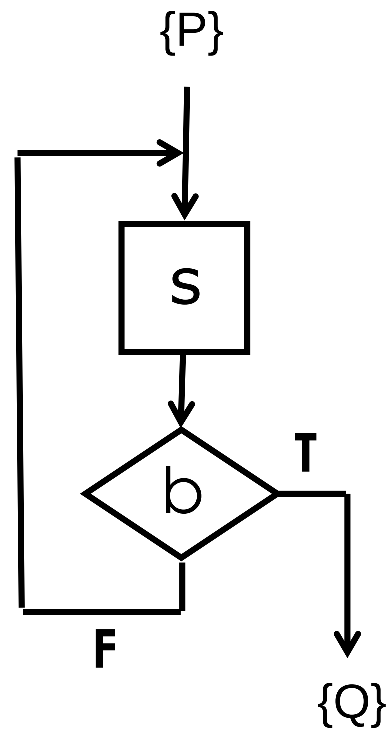
```
Found := false
Index := N
while (Index>0 & (not Found)) do
   if Key=List[Index] then
      Found := true
   else
      Index := Index-1
   end_if_else
end_while
```

$\{(Found \wedge Key=List[Index]) \vee$
$(\sim Found \wedge \forall\ 1 \leq i \leq N \bullet Key \neq List[i])\}$

$I = (Found \wedge Key=List[Index]) \vee$
$(\sim Found \wedge \forall\ Index < i \leq N, Key <> List[i])$

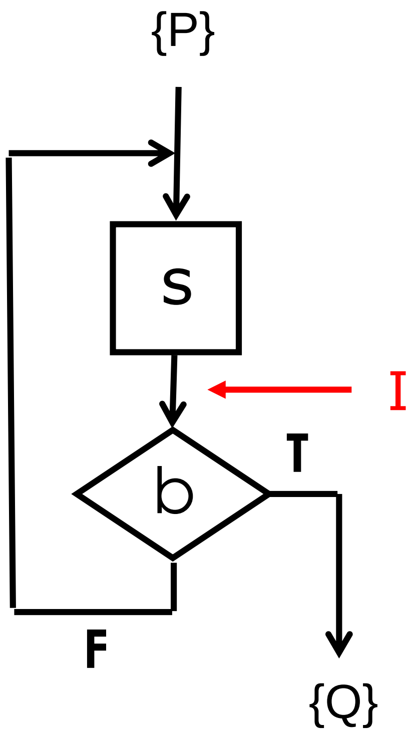
4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)

4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)



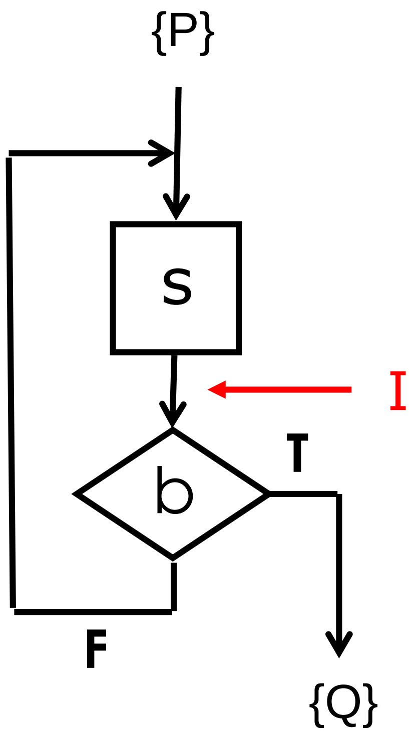
{P} repeat s until b {Q}

4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)



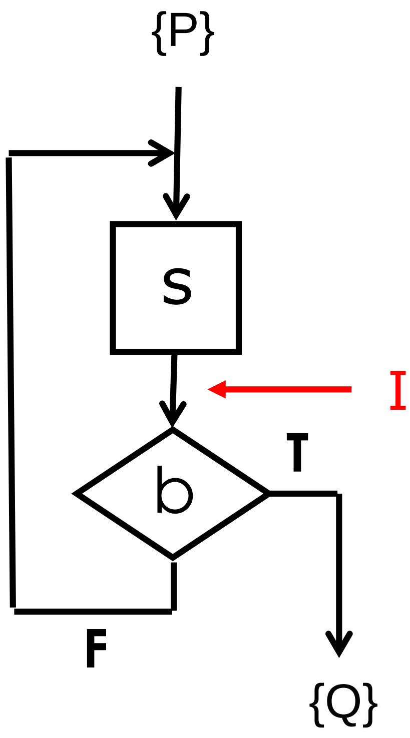
$$\frac{\qquad\qquad\qquad}{\{P\}\ \text{repeat } s \text{ until } b\ \{Q\}}$$

4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)



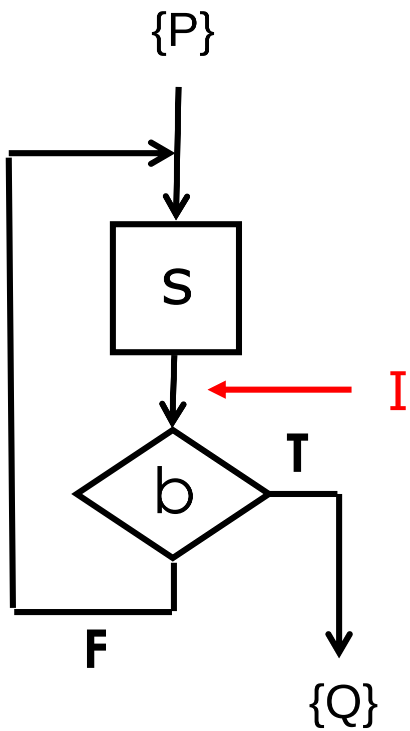
$$\frac{\{P\}\ s\ \{I\},\qquad\qquad\qquad\qquad}{\{P\}\ \text{repeat}\ s\ \text{until}\ b\ \{Q\}}$$

4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)



$$\frac{\{P\}\ s\ \{I\},\ \{I \land \sim b\}\ s\ \{I\},}{\{P\}\ \text{repeat } s \text{ until } b\ \{Q\}}$$

4. Prove the following assertion using a suitable Rule of Inference for the Repeat_Until-Loop. Clearly state the Rule of Inference and show all steps. (Hint: Do NOT include P => I as an antecedent in your rule.)



$$\frac{\{P\}\ s\ \{I\},\ \{I \land \sim b\}\ s\ \{I\},\ (I \land b) \Rightarrow Q}{\{P\}\ \text{repeat } s \text{ until } b\ \{Q\}}$$

$\{N \geq 1 \wedge \text{iorder}\}$ (where iorder $= \forall\ 1 \leq i < N \bullet List[i] \geq List[i+1]$)

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

$\{(Found \wedge Key=List[Index]) \vee (\sim Found \wedge \forall\ 1 \leq i \leq N \bullet key \neq List[i])\}$

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

P

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

{(Found ∧ Key=List[Index]) ∨ (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

$\{N \geq 1 \land \text{iorder}\}$ (where $\text{iorder} = \forall\ 1 \leq i < N \bullet \text{List}[i] \geq \text{List}[i+1]$)

```
      First := 1
      Last := N
      Found := false
P
      repeat
         Index := (First + Last) div 2
         if Key=List[Index] then
            Found := true
         else
            if Key<List[Index] then
               First := Index+1
            else
               Last := Index-1
            end-if-else
         end-if-else
      until (Found or First>Last)          I
```

$\{(\text{Found} \land \text{Key}=\text{List}[\text{Index}]) \lor (\sim\text{Found} \land \forall\ 1 \leq i \leq N \bullet \text{key} \neq \text{List}[i])\}$

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

P

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

I

<u>What is "I"?</u>

{(Found ∧ Key=List[Index]) ∨ (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

```
First := 1
Last := N
Found := false
repeat
    Index := (First + Last) div 2
    if Key=List[Index] then
        Found := true
    else
        if Key<List[Index] then
            First := Index+1
        else
            Last := Index-1
        end-if-else
    end-if-else
until (Found or First>Last)
```

P

<u>What is "I"?</u>

[(Found ∧ …) V
(~Found ∧ …)]
∧ ...

I

{(Found ∧ Key=List[Index]) V (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

$\{N \geq 1 \land iorder\}$ (where $iorder = \forall\ 1 \leq i < N \bullet List[i] \geq List[i+1]$)

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

P

[(Found ∧ …) V (~Found ∧ …)] ∧ iorder

I

What is "I"?

$\{(Found \land Key=List[Index]) \lor (\sim Found \land \forall\ 1 \leq i \leq N \bullet key \neq List[i])\}$

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

P

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

I

<u>What is “I”?</u>

[(Found ∧ Key=List[Index]) V (~Found ∧ …)] ∧ iorder

{(Found ∧ Key=List[Index]) V (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

P

```
First := 1
Last := N
Found := false
repeat
   Index := (First + Last) div 2
   if Key=List[Index] then
      Found := true
   else
      if Key<List[Index] then
         First := Index+1
      else
         Last := Index-1
      end-if-else
   end-if-else
until (Found or First>Last)
```

I

What is "I"?

[(Found ∧ Key=List[Index]) V
(~Found ∧ ∀ ? ≤ i ≤ ? • Key≠List[i])]
∧ iorder

{(Found ∧ Key=List[Index]) V (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

{N≥1 ∧ iorder} (where iorder = ∀ 1≤i<N • List[i]≥List[i+1])

P

```
First := 1
Last := N
Found := false
repeat
    Index := (First + Last) div 2
    if Key=List[Index] then
        Found := true
    else
        if Key<List[Index] then
            First := Index+1
        else
            Last := Index-1
        end-if-else
    end-if-else
until (Found or First>Last)
```

I

<u>What is "I"?</u>

[(Found ∧ Key=List[Index]) V (~Found ∧ ∀ i ∈ [1,First) U (Last,N] • Key≠List[i])] ∧ iorder

{(Found ∧ Key=List[Index]) V (~Found ∧ ∀ 1≤i≤N • key≠List[i])}

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

a.

$$\frac{P \Rightarrow (\sim b \wedge Q)}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

b.

$$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

…Clearly indicate whether or not the rule is **valid.** If valid, provide an assertion of the form {P} while b do S {Q} for which it *could* be used. If not valid, prove this by providing a counterexample.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

a.
$$\frac{P \Rightarrow (\sim b \land Q)}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

b.
$$\frac{\{P \land b\}\ s\ \{I\},\ \{I \land b\}\ s\ \{I\},\ (I \land \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

…Clearly indicate whether or not the rule is **valid.** If valid, provide an assertion of the form {P} while b do S {Q} for which it *could* be used. If not valid, prove this by providing a counterexample.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

a.

$$\frac{P \Rightarrow (\sim b \wedge Q)}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

The rule is **valid**, since the antecedent implies that whenever the pre-condition, P, holds, the false branch will be executed and Q holds. The rule could be employed, for example, to prove:

$$\{x=17\}\ \text{while } x<0 \text{ do } x := 0\ \{x>0\}$$

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b. $$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

…Clearly indicate whether or not the rule is **valid.** If valid, provide an assertion of the form {P} while b do S {Q} for which it *could* be used. If not valid, prove this by providing a counterexample.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b.

$$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

…Clearly indicate whether or not the rule is **valid.** If valid, provide an assertion of the form {P} while b do S {Q} for which it *could* be used. If not valid, prove this by providing a counterexample.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b.

$$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

The rule is **NOT valid**. (Why?)

.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b.
$$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

The rule is **NOT valid**. (Why?)

Question: How can this be *proven* using a *counterexample?*

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b. $$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

The rule is **NOT valid**. (Why?)

Question: How can this be *proven* using a *counterexample?*

Answer: (1) Identify a specific, concrete program of the form while b do s together with pre- and post-conditions such that **{P} while b do s {Q} does NOT hold**. (2) Identify an invariant I such that **all three antecedents of the rule DO hold**. This proves that the rule is not valid for the same reason that x=4 serves as a counterexample proving the rule: **["x is even" ⇒ x>10]** is not valid.

6. Consider the following HYPOTHESIZED rules of inference for the "while" construct:

b.
$$\frac{\{P \wedge b\}\ s\ \{I\},\ \{I \wedge b\}\ s\ \{I\},\ (I \wedge \sim b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } s\ \{Q\}}\ ?$$

The rule is NOT valid. Proof:

```
{y≠17}                    I: y=17
   while x>0 do
      y := 17
      x := x-1
   end_while
{y=17}
```

The three antecedents hold for the invariant y=17 but the consequent does not since the initial value of x may be ≤ 0 initially, in which case Q would not hold on termination.

Problem Set 5: Axiomatic Verification

Hints and Notes